پالایش کارتهای CRC جهت ارزیابی معماری نرم افزار

سیما عمادی[۱]، فریدون شمس[۲]

چکیده

سیستمهای نرم افزاری روز به روز در حال رشد هستند، لذا یکی از وظایف مهم توسعه دهندگان این سیستمها، ارائه راه حلهایی برای سادگی فرآیند توسعه و ارزیابی ویژگیهای کیفی آنها هست. معماری نرم افزار بعنوان اولین محصول در فرآیند توسعه نرم افزار، بهترین مرحله برای ارزیابی این ویژگیها تلقی می شود. معماری نرم افزار شامل مجموعه ای از مولفه ها، ارتباطات میان آنها و ویژگیهایی از مولفه هاست که از خارج قابل رویت هستند. برای نمایش این مولفه ها می توان از روشهای مختلفی مثل الگوهای معماری استفاده نمود. روش عمومی برای نمایش الگوهای معماری استفاده از کارتهای CRC و نمودارهای ترتیب است. به همین منظور در این مقاله کارتهای CRC جهت ارزیابی ویژگی کیفی، پالایش شده و سپس به دلیل اینکه این ساختار در ابزارهایی که UML را پشتیبانی می کنند، مورد استفاده قرار بگیرد، طی الگوریتمی به نمودار مولفه ها تبدیل می شود.

واژه‌های کلیدی

الگوهای معماری، کارتهای CRC، معماری نرم افزار، ارزیابی معماری، کارایی .

Refining CRC cards to evaluate software architecture

Sima Emadi, Fereidoon Shams

Abstract

Software systems are developing day after day; so one of the most important duties of developers is to find a solution to simplify their development process; and evaluate their qualitative attributes. Software architecture as the first artifact of software development process is considered as the best phase to evaluate such attributes. It comprises of software components, the externally visible properties of those components, and the relationships among them. In order to display the components, various methods such as architectural patterns can be applied; CRC cards and sequence diagrams are the common methods to use for representation of the mentioned patterns. In order to evaluate the performance of an architecture a refinement method for, CRC cards, is presented. Since this result should be used within tools supporting UML, an algorithm is provided to transfer CRC card to component diagram.

Keywords

Architecture patterns, CRC cards, software architecture, architecture evaluation, performance.

[1] مربی دانشگاه آزاد اسلامی واحد میبد- دانشجوی دکتری واحد علوم و تحقیقات- emadi@sr.iau.ac.ir
[2] استادیار دانشکده مهندسی برق و کامپیوتر - دانشگاه شهید بهشتی - f_shams@sbu.ac.ir

۱- مقدمه

امروزه کاربرد سیستمهای نرم افزاری، میزان سرمایه گذاری برای توسعه آنها، تعداد کاربران و حجم آنها روز به روز در حال رشد است. لذا توسعه سیستمهای نرم افزاری بر مبنای اصول و متدولوژیهایی که ضمن کاهش هزینه سرمایه گذاری شده، کلیه ویژگیهای مورد نیاز صاحبان سهام (اعم از نیازهای وظیفه مندی و غیر وظیفه مندی) این سیستمهای نرم افزاری را برآورده کند، ضروری است. با توجه به اینکه بررسی و ارزیابی این ویژگیها قبل از مرحله طراحی و پیاده سازی، هزینه و زمان کمتری را صرف می کند، لذا بهترین زمان برای سنجش رفتار قابل ارزیابی سیستم موقعی است که معماری نرم افزار آن سیستم ایجاد شده است. معماری نرم افزار بعنوان اولین محصول، نقش مهمی را در توسعه سیستمهای نرم افزاری پیچیده بازی می کند و با کمک آن می توان رفتار قابل ارزیابی سیستم یعنی صفات کیفیتی همچون امنیت، قابلیت اطمینان، قابلیت استفاده، قابلیت تغییر، ایستایی و کارایی را مورد سنجش قرار داد [۱و ۲].

برای توصیف معماری می توان از زبانهای توصیف معماری[۱]، MSC[۲] یا نمودارهای UML استفاده کرد. در این توصیفات، مولفه های سیستم و رابطهای بین مولفه ها به نمایش گذاشته می شوند. بعنوان مثال در [۳] مولفه ها، ارتباط بین آنها، نقشها و محدودیتهای آنها به کمک یک ADL بنام OPNADL[3] و در [۴] رفتار پویای سیستمها با ADLMAS[4] توصیف شده است. در [۵] مدل اجرایی نرم افزار و سیستم به کمک نمودارهای موارد کاربری، استقرار و ترتیب، در [۷] معماری سیستمهای خادم و مخدوم با نمودارهای موارد کاربری، کلاس و همکاری و در [۸] نیز با استفاده از UCM[۵] توصیف گردیده است. همچنین برای بیان مولفه ها و ارتباطات میان آنها در یک معماری می توان از الگوهای معماری استفاده کرد. الگوهای معماری ارائه کننده راه حلی برای مسائل تکرارشونده هستند و عناصر تشکیل دهنده آنها زمینه[۶]، مشکل[۷] و راه حل[۸] هست [۹]. برای نمایش معماری نرم افزار مبتنی بر الگوها نیز می توان از نمودارهای UML استفاده نمود [۶ و ۷]. بعلاوه روش کلی برای نمایش این نوع معماریها، استفاده از کارتهای CRC و نمودارهای ترتیب می باشد. در این روش کلیه الگوهای معماری موجود به راحتی نمایش داده می شوند [۹]. لذا برای نمایش معماری با استفاده از کارتهای CRC بایستی نشان داده شود که این کارتها قابلیت نمایش مولفه ها را نیز دارند.

همانطور که بیان شد معماری نرم افزار بهترین محصول برای ارزیابی نیازهای وظیفه مندی و غیر وظیفه مندی تلقی می شود، لذا برای ارزیابی این نیازها، اطلاعات مربوطه در قالب پارامترهایی به آنها اضافه می شود. در روشهای موجود، این اطلاعات در قالب UML profile یا حاشیه نویسی هایی به نمودارهای UML اضافه شده است [۶]. ولی چون کارتهای CRC و نمودارهای ترتیب، روش کلی برای نمایش معماری مبتنی بر الگوها ست، در این مقاله، به این کارتها، اطلاعات مورد نیاز برای ارزیابی صفت کیفیتی کارایی اضافه می گردد. برای رسیدن به این هدف بایستی کارتهای CRC پالایش شوند.

در ادامه ساختار مقاله بدین شرح بیان می گردد: در بخش ۲، کارتهای CRC مرور گردیده، در بخش ۳، پالایشهای انجام شده بر روی کارتهای CRC مرور می شود. در بخش ۴، کارتهای CRC جهت ارزیابی کارایی، پالایش می شود. در بخش ۵، یک مثال کاربردی در زمینه سیستم تجارت الکترونیکی برای کاربرد روش بیان شده در این مقاله مطرح می شود و در بخش ۶، نتیجه گیری بیان می گردد.

۲- کارتهای CRC

کارتهای CRC[۹] بوسیله Kent Beck و Ward Cunnigham جهت کمک به طراحان و برنامه نویسان غیر شی گرا در شناسایی اشیاء معرفی شدند [۱۱ و ۱۲]. هر کارت CRC از سه قسمت تشکیل شده است، اولین بخش نام کلاس است و مسئولیتها و همکاران آن کلاس، بخشهای بعدی آن هستند. شکل ۱ نشان دهنده یک کارت CRC است. این کارتها، روشی برای تعریف مسئولیتهای هر کلاس، نحوه تعامل کلاسها با یکدیگر و چگونگی فراهم نمودن سرویسها بوسیله آن کلاس هستند. بنابراین در یک

سیستم، برای هر کلاس بایستی یک کارت CRC رسم نمود [۱۱ و ۱۲].

Class name:	
Responsibilities	**Collaborators**

شکل ۱- کارت CRC

۲-۱- مزایای کارتهای CRC

کارتهای CRC مزایایی دارند که باعث استفاده آنها در توصیف سیستمهای مختلف می شود. این مزایا عبارتند از:

۱- نمایش تصویری کلاسهای سیستم
۲- روشی برای کمک به شناسایی کلاسها و عملیات[۱۰] آنها در طراحی شی گرا
۳- نقطه ای مناسب برای شروع تحلیل
۴- قابل جابجا بودن آنها توسط ابزار پشتیبان این کارتها
۵- قابلیت شبیه سازی رفتار سیستم (مرور سناریو) به کمک آنها
۶- بیان معماری یک سیستم به کمک آنها
۷- امکان کار گروهی
۸- پشتیبانی ابزارهای مختلف از این کارتها (Rational CRC و Vp_UML)
۹- عدم وجود نحو جهانی قابل قبول برای آنها و در نتیجه قابل تغییر بودن برای هر کاربرد
۱۰- می توانند بعنوان ورودی برای یک متدولوژی فرمال یا شبه فرمال در نظر گرفته شوند.

۳- مروری بر کارهای انجام شده جهت پالایش کارتهای CRC

با توجه به قابل تغییر بودن کارتهای CRC و همچنین جهت افزایش کاربرد آنها می توان تغییراتی روی آنها اعمال نمود. در اولین پالایش انجام شده، به این کارتها سه بخش super class، subclass و attribute اضافه شده است (شکل ۲).

Class name:	
Super classes:	
Sub classes:	
Attributes:	
Name	description
Responsibilities	**Collaborators**

شکل ۲- یک کارت CRC پالایش شده

بعلاوه می توان از این کارتها برای نمایش نقشها[۱۱] بجای کلاسها استفاده کرد [۱۳]. در اینحالت پالایش انجام شده بر روی این کارتها بصورت محدود کردن ترتیب اجرای مسئولیتها می باشد که محدودکردن ترتیب اجرای مسئولیتها از طریق عبارات مسیر[۱۲] انجام می شود. ترتیب اجرای مسئولیتها نیز توصیف کننده رفتارهای ممکن یک نقش است. در این کارتها، ترتیب اجرای مسئولیتها با استفاده از ساختارهای ترتیب، شرطی، تکرار و همزمانی مسئولیتها صورت می گیرد که برای هریک از این موارد از یادداشتهایی[۱۳] استفاده می شود. شکل ۳ یک کارت پالایش شده با مفاهیم بالا را نشان می دهد. همانطور که در شکل ۳ نشان داده شده است، در این کلاس اپراتور + تکرار مسئولیتها یا رفتار تکراری یک نقش را نشان می دهد، پرانتز مشخص کننده مجموعه مسئولیتهایی است که بایستی تکرار شوند و ; ترتیب اجرای مسئولیتها را نشان می دهد.

Class name:producer	
Responsibilities	**Collaborators**
(Produce_item; Put_item)+	_____ buffering

شکل ۳- یک کارت پالایش شده برای نقشها [۱۳]

۴- پالایش کارتهای CRC به منظور ارزیابی کارایی و تبدیل به نمودار مولفه در UML

همانطور که بیان شد، بهترین مرحله جهت ارزیابی ویژگیهای نرم افزار، مرحله معماری نرم افزار است. معماری نرم افزار شامل تعدادی مولفه، ارتباطات میان آنها و ویژگیهایی از آنها است که از خارج قابل رویت باشند. مولفه در برنامه نویسی شی گرا و فن آوری توزیعی یک سازه بلوک قابل استفاده مجدد است که می تواند با مولفه های دیگر برای ایجاد یک کاربرد ترکیب شود. برای رسیدن به این هدف، مولفه ها یک یا چند واسط را پیاده سازی می کنند تا محدودیتها و چگونگی تعامل مولفه ها در آنها توصیف شود. مولفه های معماری بصورت صفات، عملگرها و محدودیتها توصیف می شوند [۱۴]. توسعه نرم افزارهای مبتنی بر مولفه ها بسیار شبیه توسعه شی گرایی است. اگرچه توسعه مبتنی بر مولفه در سطح بالاتری از توسعه شی گرایی قرار دارد اما نگرش مجردتری از سیستمهای نرم افزاری را ارائه می دهد و مولفه ها، موجودیتهای بزرگتری نسبت به اشیا و کلاسها هستند. با این وجود شباهتهایی بین مولفه ها و اشیا و کلاسها وجود دارد. نویسندگانی همچون Orfali و دیگران معتقد هستند که یک مولفه مثل یک کلاس است منتهی با ویژگیهای اضافه تر [۱۵ و ۱۶]. سیستمهای مبتنی بر مولفه بر پایه ۴ اصل محصور سازی، چند ریختی، انقیاد دیر هنگام و ایمنی توسعه می یابند که این ۴ اصل با اصول شی گرایی مطابقت دارد[۱۷]. از طرفی مولفه ها مثل کلاسها از هم مستقل بوده و مستقیما به یکدیگر دسترسی ندارند. جهت دسترسی یک مولفه به مولفه های دیگر، یک ثبت مرکزی کلیه اطلاعات مربوط به مولفه ها و روابطشان را نگهداری می کند. هر وقت یک مولفه عملیاتی را صدا بزند، از ثبت مرکزی، واسط مورد نیاز درخواست شده و این درخواست، اشاره گری به مولفه مورد نظر را بر می گرداند. همچنین شبیه کلاسها، پیاده سازی و ساختار داخلی مولفه ها نیز از مولفه های دیگر مخفی است. اما تفاوت این دو موجودیت در مفهوم ارث بری است که در اشیا وجود دارد ولی مولفه ها از این مفهوم استفاده نمی کنند. بنابراین کلا نمی توان به ساختار داخلی مولفه ها برخلاف اشیا دسترسی داشت [۱۷].

در کارهای قبلی، کارتهای CRC جهت نمایش نقشها پالایش شدند. همچنین نشان داده شده که نقشها ویژگیهایی مشابه کلاسها اما در سطح انتزاعی بالاتری داشته و به همین علت نقشها را نیز می توان با این کارتها نمایش داد[۱۳]. همچنین با توجه به شباهتهای بیان شده بین کلاس و مولفه در معماری می توان از کارتهای CRC برای نمایش مولفه ها و معماری آنها استفاده کرد. بنابراین هر کارت با همان ساختار قبلی، نشان دهنده یک مولفه در معماری، مسئولیتها در آن نشان دهنده وظایفی از هر مولفه است که از خارج مولفه قابل رویت است و همکاران آن نشان دهنده ارتباط بین آنها است. در این پالایش در صورتیکه نام مسئولیت با نام تابع در مولفه همکار یکی نباشد، نام تابع بعنوان آرگومان نام همکار ذکر می شود. با این توصیفات، کارتهای CRC ، پالایش شده و برای نمایش معماری استفاده می شوند. از طرفی معماری بهترین مرحله برای ارزیابی ویژگیهای کیفی سیستم نرم افزاری است، لذا کارتهای CRC نیز بایستی این قابلیت را داشته باشند که این ویژگیها را در آن مورد ارزیابی قرار داد. به همین منظور به این کارتها پارامترهای مربوط به این ویژگیها اضافه می شود.. همچنین به دلیل عدم پشتیبانی ابزارهای موجود از این کارتهای پالایش شده، طی الگوریتمی این کارتها به نمودار مولفه ها در UML تبدیل می شوند. در ادامه ابتدا نحوه اضافه کردن پارامترهای کارایی به این کارتها و سپس تبدیل به نمودار مولفه توضیح داده می شود.

۴-۱- الحاق پارامترهای کارایی به کارتهای CRC

برای قابل استفاده بودن این کارتها در ارزیابی کارایی، بایستی پارامترها و اطلاعات مربوط به کارایی به مولفه های موجود در این کارتها اضافه شود. جهت اضافه کردن اطلاعات مربوط به کارایی از UML SPT profile ارائه شده در UML 2.0 استفاده می شود، که بصورت کلیشه ها[۱۴] و برچسبهایی[۱۵] اضافه

می شوند[۱۸]. همچنین به دلیل ارتباط نزدیک بین کارتهای CRC با نمودار مولفه ها در UML، کاربردهای آنها نیز یکسان است. حاشیه نویسی[۱۶] کارتهای CRC، فراهم کننده اطلاعاتی در رابطه با پارامتری نمودن مراکز سرویس در یک شبکه صف یا شبکه پتری است. مثل نوع مرکز سرویس(بعنوان مثال سرویس دهنده هایی با صف انتظار یا تاخیر)، نرخ سرویسهایی که آنها فراهم می کنند و سیاست زمان بندی که آنها جهت استخراج کارها از صف استفاده می کنند.

از کلیشه <<PAhost>> برای حاشیه نویسی مولفه موجود در هر کارت استفاده می شود. این کلیشه جهت مدلسازی یک منبع فعال بکار می رود. در اینجا فرض می شود که هر مولفه روی یک وسیله فعال[۱۷] منطقی مستقر شده است. بنابراین یک ارتباط قوی بین مولفه های نرم افزاری و منابع فعالی که آنها را پردازش می کند، وجود دارد. در صورتیکه فرض شود تمام وسایل منطقی قدرت پردازشی یکسانی دارند، برچسبی برای آنها مشخص نمی شود به استثناء برچسب PAschedpolicy که نشان دهنده سیاست زمان بندی صف انتظار مولفه ها می باشد. در غیر اینصورت در برچسب، توان پردازشی هر وسیله مشخص می شود. همچنین به هر یک از همکاران در کارت که بیان کننده ارتباط بین مولفه ها هستند، کلیشه <<PAstep>> الحاق می شود که دارای برچسبهای PAdelay یا PAdemand است. این کلیشه نشان دهنده زمانی است که یک مولفه برای انجام وظایفش نیاز دارد و برچسبهای آن کل زمان سرویس دهی مورد نیاز یک مولفه، جهت سرویس دهی به یک درخواست را نشان می دهند.

۴-۲- تبدیل کارتهای CRC به نمودار مولفه ها

همانطور که بیان شد، هر کارت نشان دهنده یک مولفه، مسئولیتهای قابل رویت آن از خارج و ارتباط آن با مولفه های دیگر است. بنابراین درصورتیکه ابزاری از آن پشتیبانی کند به راحتی برای ارزیابی نیازمندیها و در سطوح دیگر فرآیند توسعه سیستم قابل استفاده است. امروزه ابزار VP UML 5.3 از آنها پشتیبانی می کند ولی قابلیت اضافه کردن پارامترهای کارایی در آنها درنظر گرفته نشده است [۱۹]. لذا این انگیزه مطرح می شود که جهت استفاده از آنها در مراحل بعدی فرآیند توسعه، به یکی از نمودارهای UML تبدیل شود. این کارتها ارتباط نزدیکی با نمودار مولفه ها در UML دارند. به همین علت در ادامه الگوریتمی ارائه می شود که کلیه کارتها به این نمودار تبدیل شوند. الگوریتم نگاشت کارتهای CRC به نمودار مولفه به دو قسمت تقسیم می شود :

۱- نگاشت هر کارت CRC به یک مولفه: به دلیل شباهت بیان شده بین کلاسها و مولفه ها، به ازای هر کارت یک مولفه در نظر گرفته می شود و نام مشخص شده در کارت بعنوان نام مولفه منظور شود.

۲- نگاشت مسئولیتها و همکاران آن در کارتهای CRC به ارتباطات موجود در نمودار مولفه: در هر کارت مقابل هر مسئولیت، مولفه همکار آن بیان شده است که این دو مجموعه با هم نشان دهنده نحوه ارتباط دو مولفه با یکدیگر است. از طرفی ارتباطات در نمودار مولفه به دو دسته ارتباط "فراهم می کند" و "نیاز دارد" تقسیم می شوند که ارتباط اولی با یک خط ارتباطی به همراه یک دایره و دومی با یک نیم دایره مشخص می شود. بنابراین نگاشت دو عنصر کارتهای CRC به دو ارتباط موجود در دیاگرام مولفه بدینصورت است:

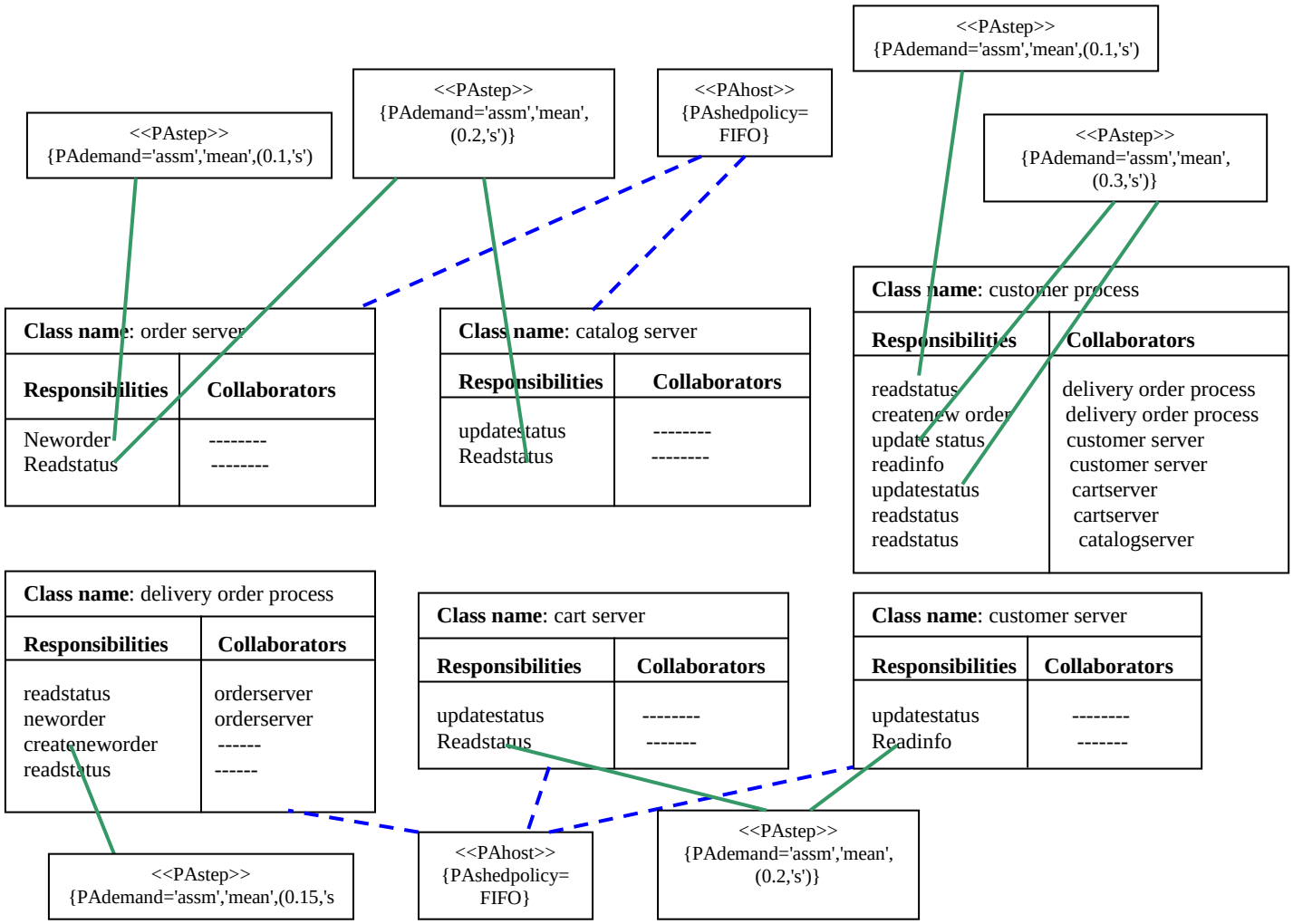
۲-۱- برای هر مسئولیتی که مقابل آن نام مولفه همکار ذکر شده باشد ارتباط "نیاز دارد" درنظر گرفته می شود و با نام مسئولیت حاشیه نویسی می شود.

۲-۲- برای هر مسئولیتی که بدون نیاز به مولفه همکار انجام می شود و فراهم کننده نیاز مولفه های دیگر است ارتباط "فراهم می کند" در نظر گرفته می شود و با نام عملیاتی که نیاز را فراهم می کند حاشیه نویسی می شود.

۲-۳- در صورتیکه حاشیه نویسی روی این دو ارتباط یکسان باشد، یکی از آنها حذف می شود.

۵- مثال کاربردی

جهت نمایش کاربرد کارتهای CRC، از آنها در نشان دادن ساختار یک سیستم تجارت الکترونیکی استفاده شده است. در سیستم تجارت الکترونیک یک کارپرداز[۱۸]، کاتالوگهایش را روی وب منتشر می کند و سفارش مشتریان را دریافت کرده و سپس این سفارشات را تحویل مشتری می دهد. برای رسیدن به این هدف، اطلاعاتی در رابطه با داده های کاتالوگ و سفارش خرید برای مشتریان بایستی نگهداری شود. مشتری نیز در صورتی می تواند درخواست سفارش کالا کند که کارت اعتباری او خالی نباشد. همچنین سیستم به مشتری اجازه می دهد تا نظارت کاملی بر وضعیت سفارش و تحویل کالای خود داشته باشد. در معماری نرم افزار این سیستم تعدادی پایگاه داده و مولفه شناسایی شده است. به هر مشتری یک customer process انتساب می شود و برای هر پایگاه داده نیز یک مولفه سرویس دهنده جهت ارتباط با آن در نظر گرفته می شود. شکل ۴ کارتهای CRC این مولفه ها را به همراه پارامترهای کارایی مورد نیاز نشان می دهد. در این شکل پارامترهای کارایی برای مولفه ها و همکاران آنها درنظر گرفته شده است. شکل ۵ نیز نمودار مولفه معادل آن را نشان می دهد. همانطور که در این شکل نشان داده شده است، پارامترهای کارایی برای مولفه ها، همان پارامترهای در نظر گرفته شده برای هر مولفه در کارت CRC است و پارامترهای در نظرگرفته شده برای ارتباطات، معادل پارامترهای همکاران مولفه در کارت هستند.

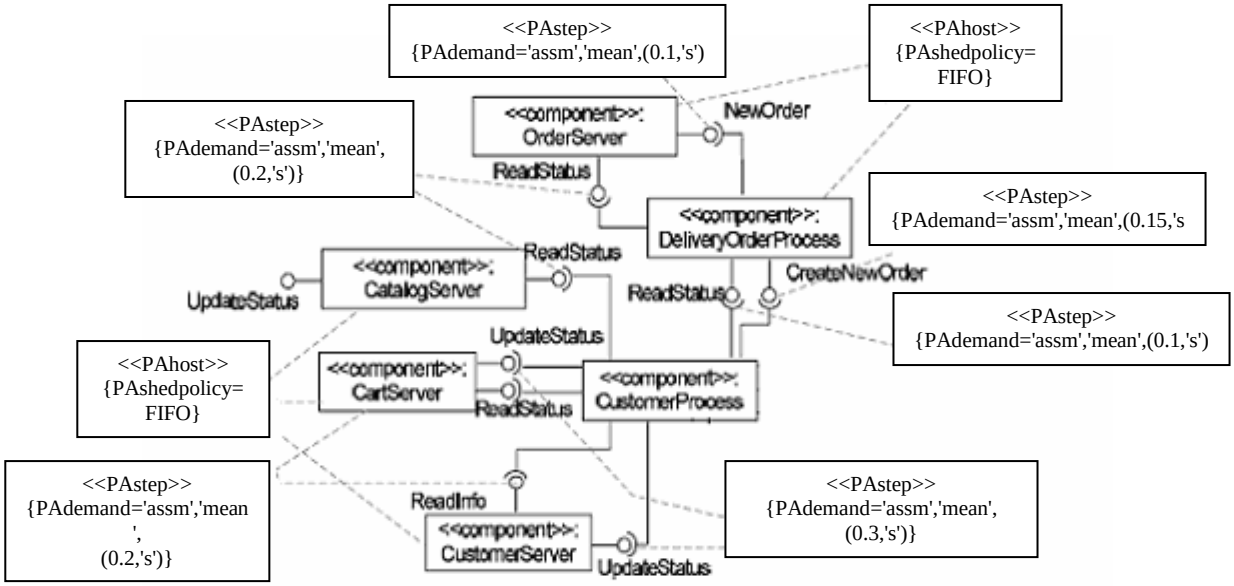


شکل ۴- کارتهای CRC مولفه های موجود در معماری به همراه پارامترهای کارایی الحاق شده به آنها

۶- نتیجه گیری

معماری نرم افزار بعنوان اولین محصول فرایند توسعه نرم افزار یکی از بهترین مراحل برای ارزیابی ویژگیهای کیفی محسوب می شود. طبق تعاریف انجام شده برای معماری نرم افزار، عناصر تشکیل دهنده این محصول، مولفه ها، ویژگیهای قابل رویت آنها و ارتباطات میان مولفه ها است که بوسیله الگوهای معماری نشان داده می شوند. الگوهای معماری، ارائه کننده یک راه حل برای یک مسئله تکرارشونده هستند و برای نمایش آنها از روشهای مختلفی استفاده می کنند. در یکی از این روشها، از کارتهای CRC و نمودار ترتیب استفاده می شود. معماری نرم افزار با کارتهای CRC که عناصر تشکیل دهنده آن، نام کلاس، مسئولیتها و همکاران آن کلاس است، ارتباط نزدیکی دارند. لذا در این مقاله با مشخص نمودن ارتباط بین کارتهای CRC و مولفه های معماری، از این کارتها جهت ارزیابی ویژگیهای کیفی استفاده می شود. به منظور ارزیابی، این کارتها، پالایش شده و پارامترهای لازم برای ویژگی کیفی کارایی به آنها اضافه گردید. در ضمن به دلیل عدم پشتیبانی ابزارهای فعلی از این نوع کارتهای پالایش شده، آنها به دیاگرام مولفه ها در UML تبدیل شدند.

در ادامه این کار انجام شده، پیشنهاد می شود که ابزارهای موجود مثل VP UML توسعه پیدا کنند تا پالایش انجام شده توسط آنها پشتیبانی شود. همچنین اضافه نمودن پارامترهای دیگر برای ارزیابی ویژگیهای کیفی به غیر از کارایی نیز قابل بررسی می باشد.



شکل ۵- نمودار مولفه معادل شکل ۴[۱۰].

مراجع

[1] Len Bass, Clements, and Kazman, “Software Architecture in Practice”, Addison Wesely, 2002.
[2] Clements, kazman and Klein, “Evaluating Software Architecture Methods and Case Studies”, Addison Wesely, 2002.
[3] Zhenhua Y, Yuanli C, "Novel Architecture Description Language based on High Level Petri Nets", IEEE, 2004.
[4] Zhenhua Yu, Zhiwu Li, "Architecture Description Language Based on Object Oriented Petri Nets for Multi Agent Systems" , IEEE, 2005.
[5] Cortellessa, V., and Mirandola, R. "PRIMA-UML: a Performance Validation Incremental Methodology on Early UML Diagrams." Science of Computer Programming.
[6] Petriu, D. C., and Shen, H. "Applying the UML Performance Profile: Graph Grammar-Based Derivation of LQN Models from UML Specifications", In Proceedings of Computer Performance Evaluation, Modeling Techniques and Tools 12th International Conference, TOOLS 2002 (London, UK, 2002), vol. 2324 of LNCS, pp. 159–177, 2002.
[7] Menasce, D. A., and Gomaa, H. "A Method for Design and Performance a Modeling of Client/Server Systems.", IEEE Transaction on Software Engineering , pp. 1066–1085, 2000.
[8] Petriu, D. C., and Woodside, C. "Software Performance Models from System Scenarios in Use Case Maps." In Proceedings of Computer Performance Evaluation, Modeling Techniques and Tools 12th International Conference, TOOLS 2002, vol. 2324 of LNCS, pp. 141–158, 2002.
[9] Frank Buschman and et.al. , “Pattern Oriented Software Architecture: a System of Pattern", 1996.
[10] Antinisca Di Marco, "Model-Based Performance Analysis of Software Architectures." PH.D Thesis, University of L’Aquila, Italy.
[11] Wilkinson, Nancy M. "Using CRC Cards: An Informal Approach to Object-Oriented Development", SIGS Books, New York, 1995.
[12] "A Laboratory for Teaching Object-Oriented Thinking" released in OOPLSA '89.
[13] Shams Fereidoon, "Modeling the Behavior of Processes using Collaborating Objects", PHD Thesis, University of Manchester, May 1996.
[14] Jun Han, "Preparing Software Components for Integration", In Proceedings of the International Workshop on Grid and Cooperative Computing, Sanya, China, December 2002, pages 679-689, Publishing House of Electronics Industry.
[15] Colin Atkinson, Thomas Kühne and Christian Bunse, "Dimensions of Component-based Development", Journal Title: 4th International Workshop on Component-Oriented Programming, 1999.
[16] Yannis Tzitzikas, "Information Systems Analysis and Design ", Lecture, University of Crete, *www.csd.uoc.gr/~hy351/2005/en*, fall 2005.
[17] Raphael Malveau, Thomas J. Mowbray, "Software Architect boot camp", 2nd Edition, Prentice Hall, 2000.
[18] ,"UML Profile for Schedulability, Performance, and Time Specification", Object Management Group, January 2005, Version 1.1, formal/05-01-02.
[19] http://www.visual-paradigm.com

---

1 - Architecture Description Language (ADL)
2 - Message Sequence Chart
3 - Object Petri Net ADL
4 - ADL for Multi Agent System
5 - Use Case Map
6 - context
7 - problem
8 - solution
9 - Class, Responsibility, Collaborator
10 - methods
11 - roles
12 - path expressions
13 - notation
14 - stereotype
15 - tag value
16 - annotation
17 - active device
18 - supplier